A MATLAB Programming Concept Inventory for Assessing First-Year Engineering Courses: a Replication of the Second Computer Science 1 Assessment

Undergraduate Thesis

Presented in Partial Fulfillment of the Requirements for the Honors Undergraduate Research Distinction in the College of Engineering of The Ohio State University

By

Ada E. Barach

The Ohio State University

2020

Thesis Committee

Dr. Krista M. Kecskemety, Advisor

Dr. Zahra Atiq

Abstract

Validated concept inventories are necessary to accurately gauge student understanding and measure classroom teaching methods. However, the First-Year Engineering (FYE) program at The Ohio State University lacks a validated assessment tool to determine student understanding of MATLAB programming concepts for first-year students. It is critical for FYE programs to have these tools available to allow the program and its instructors to determine the impacts of various teaching styles and curricula.

While there are existing concept inventories available for introductory computer science concepts, these concept inventories are programming language independent and have limitations. One such limitation is that the scores of lower-performing students tend to differ for language-independent versus language-dependent assessments more than those of high-performing students [1]. This indicates that language-independent assessments favor high-performing students. In this study, a new MATLAB-specific concept inventory, the MATLAB Computer Science 1 Assessment (MCS1), is developed by replicating a previously validated foundational computer science concept inventory, SCS1 [1, 2], for use in the FYE program at Ohio State. The goals for this project are to create an isomorphic[1] copy of SCS1 that is specific to MATLAB, conduct think-aloud interviews with first-year engineering students to determine how the assessment is being interpreted, and give both the MCS1 and SCS1 assessments to first-year students. These data will then be used in a future project to validate MCS1.

---

[1] As defined by Parker et. al, an isomorphic question "is created by maintaining the content area for a question as well as the style used to ask the question, but altering the word problem, variables, and answer choices." [12]

The result of this study is assisting in successfully developing the MCS1 concept inventory by replicating SCS1. Think-aloud interviews were conducted with six first-year students. The MCS1 and SCS1 assessments were given to a combined total of 724 FYE students in autumn 2019. Preliminary data demonstrates a statistically significant difference between SCS1 and MCS1 scores, indicating that MCS1 cannot be validated against SCS1. As a result, a future study is required to validate MCS1 on its own.

MCS1 has the potential to impact thousands of students enrolled in FYE courses at Ohio State, as well as at other universities, by normalizing the assessment process. Further, this assessment can be incorporated into the curriculum as a benchmark which can then be used by faculty and administrators to make informed decisions about the curriculum based on student retention of key concepts.

Acknowledgements

A sincere thank you to my research advisor, Dr. Krista Kecskemety, for guiding me through the research process and providing feedback and direction throughout this project. Her mentorship and support has been invaluable.

I would also like to thank Dr. Zahra Atiq for her participation in my defense committee. She has welcomed my ideas with enthusiasm and provided thoughtful comments and feedback on my thesis.

Table of Contents

List of Tables

List of Figures

Introduction

An important part of any introductory college-level course is to accurately measure the students' understanding of concepts. Many sciences, such as physics, astronomy, biology, and geosciences employ the use of validated concept inventories for this purpose [3, 4, 5, 6, 7]. However, computer science has few validated assessments available to instructors [8, 9, 10]. A concept inventory is "an outline of core knowledge and concepts for a given field and a collection of multiple-choice questions that are designed to probe student understanding of these fundamental concepts" [8]. It is important to note that concept inventories differ from a summative assessment (say, a final exam for a college course) in four fundamental ways, as described by Sands et. al:

(1) Unlike summative assessments, concept inventories are meant to be taken multiple times and to provide a benchmark against which student understanding can be compared.
(2) It is expected that students have not studied to take a concept inventory assessment as this would obfuscate which concepts students have mastered and which concepts students have committed to short-term memory.
(3) Concept inventories should be designed to test broad concepts, not specific definitions or mathematical formulas.
(4) Scores on summative assessments typically inform significant decisions regarding students, such as a final term grade. [11]

The Foundational CS1 assessment (FSC1), built by Allison Tew, was the first validated assessment tool for introductory-level programming concepts [1]. FSC1 was then replicated by Parker et al. to produce the Second CS1 assessment tool (SCS1) which is an isomorphic copy of FCS1 [12]. This isomorphic copy assesses the same concepts and uses the same question style as FCS1 but has different wording and answer choices [12]. While both of these concept inventories are computer programming language independent, students typically performed better on an associated language-dependent exam [2]. Tew found that the correlation between FCS1 and its associated language-specific version was stronger for higher-performing students than lower-performing students. Tew then concludes that this is likely due to higher-performing students being better able to transfer language-specific knowledge to pseudocode. As a result, Tew's findings suggest that language-independent exams are biased toward high-performing students [1, 12].

Other concept inventories for computer science have been developed to test specific areas of computer science including algorithms, architecture, data structures, digital logic, operating systems, and recursion [10, 13, 14, 15, 16, 17, 18, 19]. However, unlike FCS1 and SCS1, these concept inventories do not test specific concept areas and thus do not apply as broadly.

Creating a validated MATLAB concept inventory for introductory computer science, called the MATLAB Computer Science 1 Assessment (MCS1), is critical to remove the potential bias towards high-performing students that was seen in SCS1. It was conjectured that this bias was introduced due to students with greater understanding being more comfortable with the pseudocode format of SCS1 and thus being better able to transfer their language-specific knowledge to the pseudocode [12]. As a result, a language-specific assessment tool that does not rely on pseudocode will likely portray concept understanding more accurately for low-performing students. MCS1 will then enable first-year engineering instructors to better evaluate student understanding, course curricula, and teaching methods.

MCS1 will provide instructors with a tool that not only allows them to better assess the knowledge of their students but to also determine areas in the teaching material that need improvement. In addition, use of MCS1 can be expanded beyond first-year engineering to assess student retention of coding concepts. For example, MCS1 could be used in a study to determine if upperclassmen are retaining coding concepts necessary to be successful in upper-level courses. A study of this nature would then allow course-specific modules to be created that would strengthen coding concepts that students tend to forget as they progress through their academic career. Overall, having a MATLAB-specific validated assessment tool will promote consistency in student assessment across the roughly 2200 students that are enrolled to the first-year engineering program annually [11] and provide the instructional teams with a tool by which the teaching methods and course material can be evaluated and improved.

Background

A Brief History of Concept Inventories

The first concept inventory, the Force Concept Inventory, developed by Hestenes et al. as a diagnostic test to measure student understanding of force-related concepts was published in 1992 [5]. Since then, the use of concept inventories to measure student understanding and classroom teaching methods has increased in popularity [20]. As of 2008, there were at least 14 published concept inventories covering physics, chemistry, astronomy, biology, and geoscience concepts [3]. By 2014, the development of concept inventories had expanded to include fields such as natural selection, genetics, and others [9].

However, concept inventories for computer science have not developed as quickly due to a unique set of challenges [9]. Almstrum et. al point out one of these challenges in their development of a computer science concept inventory for discrete mathematics:

> *"The computing field is notorious for its dependence on notations and conventions…A notation-heavy item may end up testing students both on their knowledge of the notation and their understanding of this concept." [21]*

This challenge, along with others, has inhibited the development of computer science inventories and created a critical lack of assessment tools.

In 2011, Allison Tew and Mark Guzdial created a language-independent assessment called FCS1, the Foundational Computer Science 1 assessment [1, 2]. FCS1 is a landmark computer science concept inventory as it is among the first to be applicable to a broad range of teaching methods and programming languages. However, Tew and Guzdial found that the student scores on FCS1 may be biased based on which programming language the student first learned.

Expanding on FCS1, Parker et. al replicated FCS1 to create SCS1 – the Second Computer Science 1 assessment. SCS1 is an isomorphic copy of FCS1 and was made with the goal to further enable computer science instructor use of concept inventories. Additionally, the authors of SCS1 argued that having more concept inventories (through replicating previously validated assessments) increases the likelihood that student understanding is accurately measured. While SCS1 was found to be concurrently valid with FCS1, the authors recommended improving it due to a less than acceptable internal consistency between questions. Parker et. al's Item Response Theory analysis and Cronbach's alpha value suggest that SCS1 could be improved to have questions that exhibit a better discrimination and are less difficult [12].

The development of MCS1 follows in the footsteps of FCS1 and SCS1 and aims to provide a language-specific assessment while building and improving upon the FCS1 and SCS1 assessments.

First-Year Engineering Program at Ohio State University

For this study, participants were recruited from the FYE courses in the College of Engineering (COE) at Ohio State. The Ohio State University is a large, Midwestern university that enrolled a total of 68,262 students in autumn 2019, of which 53,669 students were undergraduates [22]. The COE at Ohio State makes up approximately 18% of the student population and is the second largest college at the university [22]. Of the 8,499 students enrolled in the COE in 2019, 1,685 students were new first-year or rank 1 and 2 transfer students [23].

First-year students in the COE are required to complete two Fundamentals of Engineering (FE) courses: ENGR 1181 (1281 honors) and ENGR 1182 (1282 honors). The ENGR 1181/1281 course is typically taken in a student's first semester and is followed by ENGR 1182/1282 course in the next semester. ENGR 1181 is the standard course which focuses on technical graphics, computer-aided design, MATLAB programming, engineering design and analysis, project management, engineering ethics, teamwork, and oral and written communication [24]. The honors version of the course, ENGR 1281, teaches the same curriculum on an accelerated course and includes an introduction to C/C++ programming [23]. In Autumn 2019, 1,888 total students were enrolled in the ENGR 1181 and 1281 courses. Of these 1,888 students, 408 were enrolled in the honors course and 1480 students were enrolled in the standard course.

Upon completion of the ENGR 1181/1281 courses, students are expected to have learned basic programming principles in MATLAB. The MCS1 concept inventory aims to evaluate student understanding of these basic principles, so students enrolled in these FE courses in Autumn 2019 were recruited to participate in this study.

Scope and Contributions

Developing MCS1 was part of a larger project to create and validate a MATLAB concept inventory for use in the FYE courses at The Ohio State University. The development of MCS1 consisted of five major stages, as outlined in Figure 1. The first stage, replication, involved translating SCS1 to MATLAB and superficially modifying parts of the question stem and responses. The next two stages were to conduct think-aloud interviews to determine if students were interpreting the questions as intended and then to make any necessary revisions to MCS1. Once a final draft of MCS1 was produced, both MCS1 and SCS1 were given to current first-year students. The final stage of this project is to use the data from administering both concept inventories to conduct a validation study on MCS1.

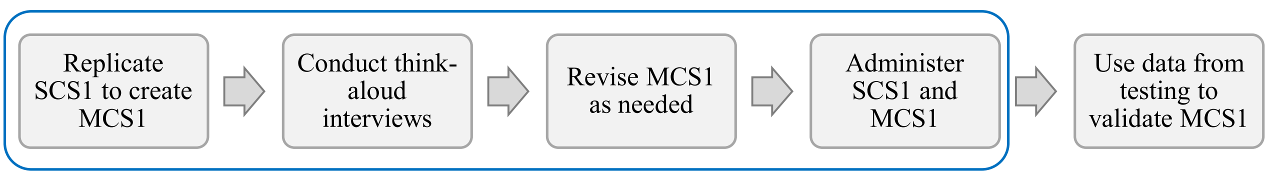

Figure 1: MCS1 Development Stages

Thesis Objectives

The goals for this thesis are as follows:

- Assist in replicating SCS1 to create a MATLAB-specific concept inventory.
- Lead and participate in conducting think-aloud interviews to determine necessary revisions to MCS1.
- Revise MCS1 based on the findings of the think-aloud interviews.
- Organize and help with the piloting of MCS1 and SCS1 assessments.

Contributions

The larger project encompassing this thesis was completed by Dr. Krista Kecskemety, me, and two other undergraduate research assistants. For this thesis, I led the development of MCS1, the think-aloud interviews, and administering the MCS1 assessment to first-year students, as outlined by the blue box in Figure 1. Thus, this document will focus on those four stages. Namely, the data collected during the testing of MCS1 and SCS1 as well as the subsequent data analysis and validation study will not be addressed.

Developing MCS1

The first step was to create an isomorphic copy of SCS1. The content, question type, format, and order of questions largely resembles SCS1 but the material has been changed to be MATLAB-specific. Each question is composed of two main components: the stem, which contains the problem statement(s), followed by the response options.

To replicate SCS1, each question stem was first altered to be MATLAB-specific. The stem may have been otherwise trivially changed to have different variable values or wording. Then, the response options were updated to have an appropriate correct response and distractors for the altered question. An example of this process is shown in Figure 2.

SCS1 (left)

Given the following code segment

```
array = [3, 6, 8, 1, 2, 0, 7, 2, 9]
i = 0
odd = 0
WHILE (i < length(array)) AND
    (array[i] != 0)
DO
        IF (array[i] % 2) == 1 THEN
                odd = odd + 1
        ENDIF
                i = i+1
        ENDWHILE
```

What are the values of the variables `i` and `odd` after the while loop completes its execution?

A. `i = 1; odd = 0`
B. `i = 5; odd = 2`
C. `i = 5; odd = 3`
D. `i = 8; odd = 4`
E. `i = 8; odd = 5`

MCS1 (right)

Given the following code segment

```
array = [2, 5, 7, 1, 4, 0, 9, 3, 6,
8];
i = 1;
even = 0;
while (i < length(array)) && (array(i)
   ~= 0)
     if (mod(array(i),2) == 0)
         even = even + 1;
     end
         i = i + 1;
end
```

What are the values of variables `i` and `even` after while loop completes its execution?

A. `i = 2; even = 1;`
B. `i = 5; even = 2;`
C. `i = 5; even = 3;`
D. `i = 6; even = 2;`
E. `i = 7; even = 2;`

Figure 2: Example Question Mapping from SCS1 (left) to MCS1 (right)

The above example shows a SCS1 question on the left and its equivalent MCS1 question on the right. The pseudocode was first converted to MATLAB and then altered to have different variable values. The response options were also updated to better reflect possible answers. In order to faithfully replicate SCS1, the question style and content area were not changed. Three questions regarding recursion in SCS1 were not included in MCS1 since recursion is not typically taught for introductory MATLAB courses and is not included in the Ohio State FYE curriculum [25, 26, 27, 28]. The goal for replicating SCS1 questions was to create questions similar in concept and style but would require students to work through the problem in its entirety again, even if SCS1 and MCS1 were taken one after the other.

MCS1 Questions and Answers

The replication of SCS1 to create MCS1 resulted in a new concept inventory that assesses student understanding of foundational computer science topics in MATLAB. MCS1 is comprised of 24 multiple choice questions across eight concepts. These concepts and a brief description are given in Table 1.

Table 1: MCS1 Question Concepts, Descriptions, and Examples

Concept	Description	Example
Arrays	Single-dimension data structures containing data of one type.	`arr = (0, 1, 2, 3, 4);`
Basics	Core knowledge not related to any other concept. This can include order of operations, basic calculations, or definitions.	`x = 2 - 3 * 5;`

For Loops	A loop structure in which the start, end, and increment values are defined.	`for 1 : 1 : 10`
		`…`
		`end`

Function Parameters	The behavior of variables defined in the definition of a function.	`function [ret] = example(`**`p1, p2, p3`**`)`
Function Return Values	The value(s) returned from a function.	`function [`**`ret`**`] = example(p1, p2, p3)`

If Statements	Conditional statements that determine which code branch to execute.	`if /* cond */`
		`…`
		`else`
		`…`
		`end`

Logical Operators	Binary operations that result in either true or false. Can also be called conditional operators.	`&,|,~`

While Loops	A loop structure in which the start and increment values are defined along with an end condition.	`int index = 1`
		`while /* cond */`
		`…`
		`end`

The above table outlines each of the eight concepts tested in MCS1. Since MCS1 has a total of 24 questions, there are three questions dedicated to testing each of these concepts. These concepts are then tested using one of the three question types:

(1) Code Completion – Filling in snippets of a code segment to accomplish a specified task

(2) Definitional – Related to the direct definition, or a fact, of a given concept

(3) Tracing – Evaluating the flow of execution for a given code segment

Of the 24 total questions, eight questions were code completion, eight were definitional, and eight were tracing. Within a question type, each of the eight questions test a different concept. Thus, all combinations of question type and concept are present in MCS1.

Research Methods

Think-Aloud Interviews

After creating MSC1, think-aloud interviews were conducted. In a think-aloud interview, a student reads through each question and explains, out loud, his/her thought process and justification for selecting an answer [29]. These interviews are then used to qualitatively determine if the questions in the assessment are being interpreted as intended. In other words, the think-aloud interviews for this study aimed to answer two questions:

(1) Are students interpreting the assessment questions the way the authors intended?
(2) Are students reasoning about the intended concepts to answer the assessment questions?

To recruit think-aloud interview participants, an email was sent to all current FYE students at Ohio State. This email contained a URL which allowed students to indicate interest in participating as well as their available times. Then, participants were selected on a first-come, first-served basis depending on their availability. Six students were interviewed: three honors students and three standard students.

The interviews lasted 60-90 minutes with each student starting at a different point in the assessment to ensure that all parts of the assessment were reviewed. Throughout the think-aloud interview, the student was prompted, as necessary, to further explain why certain answer choices were chosen or eliminated. Interviewer commentary was avoided as much as possible to reduce potential interruptions to the student's thought process. Two researchers were present at each interview: one to proctor/prompt the student if necessary and the second to take notes on student reasoning and control audio recordings. The audio recordings were transcribed for use in subsequent analyses. The analysis of these recordings aimed to be as objective as possible, but some interviewer impressions and inferences had to be made about student reasoning [29].

Interpretative Content Analysis

Once the think-aloud interviews were completed, the researcher notes and audio recordings were used to perform an interpretative content analysis. A content analysis is the process of quantitatively summarizing certain characteristics of a message or communication [30]. Specifically, an interpretative analysis focuses on the "observation of messages and the coding of those messages" [30]. In other words, each student response in the think-aloud interviews was categorized based on the codebook given below. To promote the reliability of the results, two researchers performed a content analysis for each think-aloud interview [30].

To perform the content analysis, each researcher independently evaluated a participant's response, using the transcribed audio recordings, to determine if the participant was reasoning about the intended construct and if their logic was sound. Then, using this determination, the researcher designated the appropriate category for the response using the codebook. Once the analyses were complete, the results from both researchers were checked for discrepancies. In the event of a discrepancy, the researchers who performed the analysis discussed their reasoning in front of a third party until a consensus was reached. This process of checking for discrepancies and coming to a consensus was used to promote intercoder reliability and agreement [30, 31].

The rubric used for this analysis is shown in Table 2 and was used to determine which category to assign to each student response.

Table 2: Coding Rubric for Think Aloud Interview Data [1]

Code	Description
1	Participant answered question correctly by reasoning about intended construct.
2	Participant answered question incorrectly by following common misconception or using faulty logic about construct.
3	Participant answered question correctly even though they had incorrect reasoning about construct.
4	Participant answered question correctly, however the correct answer was reached by reasoning about other conceptual content.
5	Participant answered question incorrectly due to reasoning about other constructs.
6	Participant answered question incorrectly. The wording led to confusion/incorrect answer.
8	Participant answered question incorrectly. The reasoning was incoherent and difficult to assign to any particular concept/construct.

The above rubric was developed by Allison Tew for analyzing the results of the think-aloud interviews for FCS1 [1]. Code 7 is not included here since it pertains to student transfer of knowledge to pseudocode which is not applicable to the language-specific MCS1. This rubric was chosen for use in MCS1 think-aloud data analysis to facilitate a direct comparison between FCS1 and MCS1. Though validity is not transitive (i.e., the validity of FCS1 does not translate to MCS1 since MCS1 is not a direct replication of FCS1), FCS1 and MCS1 aim to assess the same constructs and use the same question style and content. As a result, one would expect the FCS1 and MCS1 content analysis results to be similar. In this way, the comparison of FCS1 and MCS1 think-aloud interviews were used to superficially evaluate if MCS1 seemed to be testing the correct constructs. While this comparison served as a quick evaluation of MCS1, the FCS1 results cannot be used to argue any significant results about MCS1.

MCS1 Revisions

A researcher not present in the interview read through the notes and listened to the audio recordings to identify particularly confusing questions or answers. An external researcher was used to reduce the chance of bias that may result from researcher impressions or context during the interview [32]. Using the notes and recording from each interview, a revision type ("Answer Revision, "Question Revision", "Formatting", or "Typo") was assigned for each necessary revision. The compiled list of question issues and potential revisions was then evaluated by the research team to determine if a revision to MCS1 was necessary. The selected revisions were then made to MCS1.

Piloting the MCS1 and SCS1 Assessments

Following these revisions, participants were recruited to take both the SCS1 and MCS1 assessments. All students from all sections of the ENGR 1181 and ENGR 1281 first-year engineering courses were contacted via email with a URL to allow students to indicate interest in taking the assessment. The student interest data was then used to schedule a time for each student to take the assessment on a first-come, first-served basis.

The assessments were given to 724 first-year engineering students through the Ohio State University's Qualtrics online application approximately two weeks before the end of the Autumn 2019 semester. Each student was required to sign in with his/her OSU credentials. Once the students had signed in, Qualtrics randomly assigned either the MCS1 or SCS1 assessment to each student. 359 participants (49.59%) took SCS1 and 365 (50.41%) participants took MCS1. Once the assessment began, students were given 60 minutes to complete the questions before being asked to provide their demographic information and information about their prior programming experiences. The prior programming experience questions are provided in Appendix A: Programming Experience Questions. The demographic data is summarized in Tables 3 – 8.

Table 3: Summary of Participants by FYE Course

	SCS1 (n = 359)		MCS1 (n = 365)	
	Number	*Percentage*	*Number*	*Percentage*
ENGR 1181	260	76.02%	265	78.40%
ENGR 1281	82	23.98%	73	21.60%
Total	342	95.26%	338	92.60%

Table 4: Summary of Participants by Semester Enrolled in FYE Course

	SCS1 (n = 359)		MCS1 (n = 365)	
	Number	*Percentage*	*Number*	*Percentage*
Currently Enrolled	331	98.81%	334	99.11%
Summer 2019	0	0.00%	1	0.30%
Spring 2019	1	0.30%	1	0.30%
Autumn 2018	1	0.30%	0	0.00%
Summer 2018	0	0.00%	1	0.30%
Did not take the course (transfer credit, EM credit, etc.)	2	0.60%	0	0.00%
Total	339	94.43%	337	92.32%

Table 5: Summary of Participants by Age

	SCS1 (n = 359)		MCS1 (n = 365)	
	Number	*Percentage*	*Number*	*Percentage*
18 – 22 years old	336	98.53%	331	97.93%
23 – 29 years old	4	1.17%	5	1.48%
30-39 years old	1	0.29%	1	0.30%
40-49 years old	0	0.00%	1	0.30%
Total	341	94.99%	338	92.60%

Table 6: Summary of Participants by Gender

	SCS1 (n = 359)		MCS1 (n = 365)	
	Number	*Percentage*	*Number*	*Percentage*
Male	215	63.05%	219	64.99%
Female	124	36.36%	117	34.72%
Other	2	0.59%	1	0.30%
Total	341	94.99%	337	92.32%

Table 7: Summary of Participants by Race/Ethnicity

	SCS1 (n = 359)		MCS1 (n = 365)	
	Number	*Percentage*	*Number*	*Percentage*
American Indian or Alaskan Native	0	0.00%	1	0.28%
Asian/Pacific Islander	70	19.55%	83	22.99%
Black or African American	19	5.31%	11	3.05%
Hispanic American	14	3.91%	14	3.88%
White/Caucasian	248	69.27%	245	67.87%
Multiple Ethnicity/Other	7	1.96%	7	1.94%
Total	358	99.72%	361	98.90%

Table 8: Summary of Participants by Primary Language

	SCS1 (n = 359)		MCS1 (n = 365)	
	Number	*Percentage*	*Number*	*Percentage*
Chinese	14	4.13%	18	5.33%
English	293	86.43%	287	84.91%
French	1	0.29%	0	0.00%
Korean	1	0.29%	5	1.48%
Russian	1	0.29%	2	0.59%
Spanish	5	1.47%	2	0.59%
Vietnamese	1	0.29%	2	0.59%
Other/Multiple	23	6.78%	22	6.51%
Total	339	94.43%	338	92.60%

After completing the assessment, an automatic email was sent to each student containing a breakdown of his/her scores for each question category. With IRB approval, students who completed the assessment were awarded extra credit in his/her engineering course. Additionally, students were incentivized to participate since their score breakdown provided information on which areas to study in preparation for their final exam.

Results and Analysis

The following section discusses the think-aloud interview results as well as the chronological analysis of these data. The first subsection details the qualitative review of the transcribed audio recordings and researcher notes to determine needed revisions to MCS1. The following subsection explores the aggregate score data for each question. The last subsection examines the results from performing the interpretative content analysis.

Revisions to MCS1 based on Think-Aloud Interviews

The first draft of MCS1 was given to six students – three honors and three standard– to gain insight into how students were interpreting the MCS1 questions and to catch areas of confusion resulting from the test itself. Think-aloud interview participants were selected on a first-come, first-served basis from the responses to the recruitment email.

After completing the interviews, the audio recordings and researcher notes were used to create a list of revisions containing the revision type, the question needing revision, the number of participants who commented on the issue, and the question revision itself. A portion of this data can be seen in Figure 3. The full list can be found in Appendix B. Think-Aloud Interview Revisions.

#	Revision Category	Issue	# Participants Reporting	Resolved?	Notes
1	Question Revision	Removed "at the beginning" from choice A	2	Yes	
2					
3					
4	Question Revision	Update to have E be the correct answer	4	Yes	
4	Question Revision	Remove "of 4 characters" in code comment.	Researcher	Yes	
4	Typo	Make "not" in question bold and italicized to match other questions	Researcher	Yes	
5	Question Revision	Inconsistency between double quotes in question vs. single quotes in the function definition	1	Yes	
5	Typo	Removed parentheses in if-loop to be more consistent with other questions	Researcher	Yes	
5	Formatting	Change "concatentation" in question to be default font.	Researcher	Yes	
6	Question Revision	Conditional operators is defined in SCS1 pseudo code but not in MCS1	3	Yes	Add description in question

Categories
Answer Revision
Question Revision
Formatting
Typo

Figure 3: Example Think-Aloud Interview Data

In analysis of the think-aloud interview data, each issue found with the assessment was categorized in one of four ways:

1. Answer Revision – The response options for a question use poor wording or are incorrect.
2. Question Revision – The stem is poorly worded or incorrect. This category also includes inconsistency in writing style.
3. Formatting – The formatted appearance of the stem and/or response options is either misleading, inconsistent with the rest of the assessment, or both.
4. Typo – A typographical error exists in the stem and/or response options.

To help determine the severity of a needed revision, the number of participants who recorded the error was noted. In some cases, these errors did not affect participants but were noticed by members of the research team.

Think-Aloud Interview Participant Performance on MCS1

As the think-aloud interviews are intended to provide insight to how students will perform on the assessment, the research team also reviewed the scores of think-aloud interviews. Table 9 details the response and accuracy rates for the think-aloud interview assessment results.

Table 9: Think-Aloud Interviews: Response Rate and Accuracy by Question

Question	Number of Responses	Response Rate (%)	Number of Correct Responses	Response Accuracy (%)
1	5	83.33	5	100.00
2	5	83.33	4	80.00
3	5	83.33	5	100.00
4	4	66.67	3	75.00
5	6	100.00	1	16.67
6	5	83.33	3	60.00
7	5	83.33	2	40.00
8	4	66.67	4	100.00
9	4	66.67	1	25.00
10	3	50.00	2	66.67
11	5	83.33	4	80.00
12	4	66.67	1	25.00
13	6	100.00	4	66.67
14	4	66.67	3	75.00
15	6	100.00	5	83.33
16	5	83.33	5	100.00
17	6	100.00	6	100.00
18	6	100.00	1	16.67
19	6	100.00	1	66.67
20	5	83.33	5	100.00
21	6	100.00	5	83.33
22	6	100.00	5	83.33
23	4	66.67	3	75.00
24	5	83.33	1	20.00

Looking at the above table, a majority of participants (who answered the question) correctly answered 16 of the 24 questions in MCS1. Further, only 5 of the 24 questions had a correct response rate of less than or equal to 25%. Participant accuracy by question is restated in the following figure which illustrates the number of questions that were correctly answered by participants. Each section of the chart corresponds to an accuracy range, i.e., the green section indicates the number of questions for which 76-100% of the responding participants selected the correct answer, yellow corresponds to the questions for which 51-75% of participants selected the correct answer, etc.

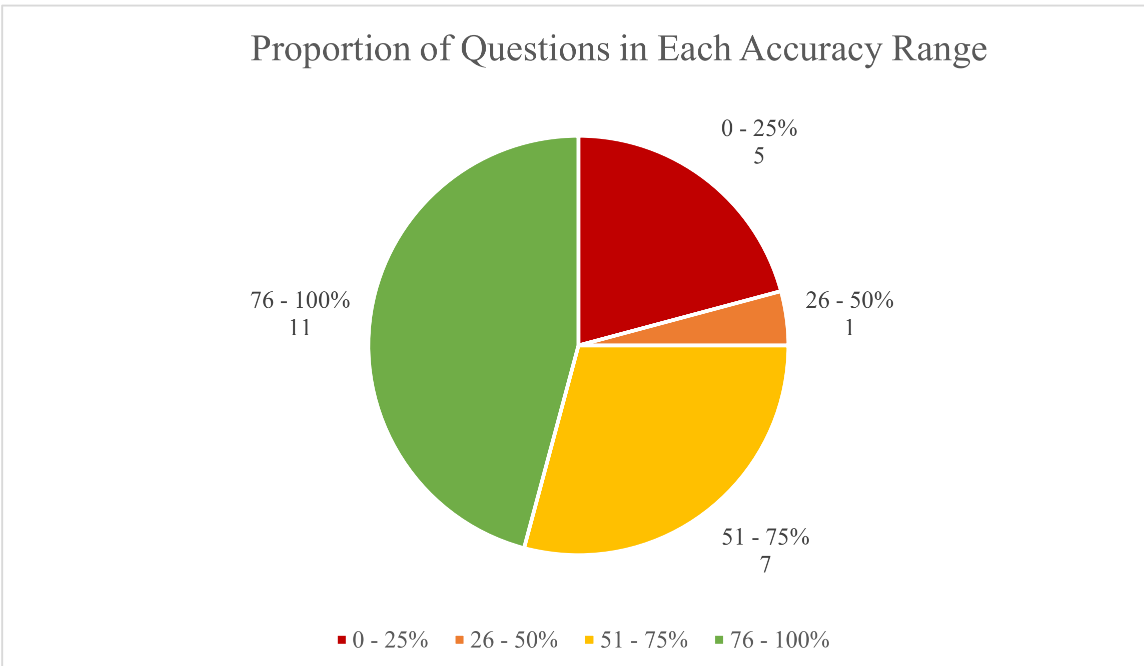

Figure 4: Number of Questions in Each Accuracy Range

The above figure splits up MCS1 think-aloud participant response data by the percentage of participants who correctly answered a given question. A plurality of questions had the highest percent of correct responses, meaning that over 75% of the participants who answered these 11 questions selected the correct answer. Similarly, a majority of the questions had at least half of the participants who answered the question select the correct answer.

Interpretative Content Analysis for Think-Aloud Interview Responses

An interpretative content analysis was also performed on the think-aloud interview results [30]. For each interview, two researchers separately categorized each participant response to a question. To maintain an independent analysis, each researcher coded students' responses without knowledge of the other researcher's categorization. Any discrepancies between the two analyses were discussed until a consensus regarding the correct category was reached. The results from the content analysis were used to determine if MCS1 questions were successfully testing the intended construct by evaluating the percentage of participant responses in categories 1 and 2. Further, this analysis served as a pilot for the future testing of MCS1. The response categories and the percent of responses in each category is shown in Table 10.

Table 10: Rubric and Results for Coding Student Response Data

Code	Description	%
1	Participant answered question correctly by reasoning about intended construct	62.71
2	Participant answered question incorrectly by following common misconception or using faulty logic about construct	27.12
3	Participant answered question correctly even though they had incorrect reasoning about construct	3.39
4	Participant answered question correctly, however the correct answer was reached by reasoning about other conceptual content	5.08
5	Participant answered question incorrectly due to reasoning about other constructs	2.54
6	Participant answered question incorrectly. The wording led to confusion/incorrect answer.	0.85
8	Participant answered question incorrectly. The reasoning was incoherent and difficult to assign to any particular concept/construct.	0.00

Categories 1-3 in Table 10 indicate that participants were reasoning about the correct concept when determining their answer. Categories 4-6 discuss scenarios where participants are reasoning about a concept other than the one intended. Lastly, Category 8 handles participant responses that are not easily assigned to one of the other categories.

Since think-aloud interviews are used to determine if an assessment is testing the intended concepts, responses in Categories 1 and 2 are of particular interest. A response that is categorized under Category 1 shows that the participant is determining the correct answer to the question by using sound logic about the concept that the question is meant to test. An example of such a response is given below:

> *"If (x & y) is true then that half of the OR is true and since it's an OR statement, only one has to be true so, um, the entire second expression evaluates to true."*

On the other hand, Category 2 indicates that the student reasoned about the intended construct but reached an incorrect answer due to flawed logic. An example of a Category 2 response is given below:

> *"The order of the inputs to the function call is unimportant. That's true because the function definition declares how the inputs are used. So if I put in x, y, and z to calculate, it's the same thing as y, x, and z depending on how I define them in the function."*

In the above quote, the participant is reasoning about the behavior of function parameters which is the intended construct. However, their logic in the first sentence is flawed since in MATLAB, the order of parameters when making a function call is important. In other words, making the function call `calculate(x, y, z)` may result in different behavior than `calculate(y, x, z)`. It should be noted that the participant's reasoning in the second sentence is generally correct but does not apply to the question since the definition of the variables within the function was given as part of the question stem.

For the MCS1 think-aloud interviews, 89.83% of responses were either Category 1 or 2, indicating that a large majority of participants were reasoning about the intended constructs. This percentage is higher than the corresponding percentage (83%) of Category 1 and 2 responses found in Tew's development of FCS1 [1]. As a result, MCS1 may be better gearing its questions towards the intended construct than FCS1. It was also found that 5.08% percent of MCS1 think-aloud interview participants reached the correct answers by reasoning about other concepts while FCS1 reported 0% of its responses in this category. This difference is likely due to differing interpretations of the category description by the MCS1 and FCS1 teams. Overall, the results from the MCS1 think-aloud interviews indicate that students are interpreting the questions as intended and are, for the most part, reasoning about the correct constructs.

Though out-of-scope for this document, the data collected from the piloting of the MCS1 and SCS1 assessments will be statistically analyzed to determine if MCS1 is a valid assessment. Additionally, the think-aloud interview results also provide data necessary for the future validation study on MCS1.

Conclusion

The development of MCS1 provides instructors in FYE programs with a validated assessment that can be used to evaluate student understanding, teaching methods, and course curricula. It is hoped that MCS1 will be incorporated into the FYE program as an additional tool for instructors.

To protect the validity of MCS1, the questions and answers are not provided here. A major concern for any validated assessment is the saturation of its questions and answers. If the questions and answers are easily locatable by students, the accuracy with which MCS1 can measure student understanding is reduced since it is possible that students could have previously seen the questions or answers.

Additional Applications

Outside of MCS1's application within the Engineering Education department at The Ohio State University, MCS1 has the potential to impact thousands of other first-year engineering students across the United States. This concept inventory is not specific to Ohio State and can thus be used as a benchmark or evaluation for any first-year engineering program that teaches MATLAB.

Future Work

Following this project, a full data analysis and validation study for MCS1 is recommended. First, in alignment with the goal of this project, the participant scores from the MCS1 and SCS1 assessments given in autumn 2019 will need to be analyzed to determine if there is a correlation in student performance between the two assessments. This correlation, if it exists, as well as an Item Response Theory analysis, can then be used to argue that the MCS1 assessment is valid since it correlates to the SCS1 assessment which has been previously validated. Further, due to the amount of data recorded, a standalone validation study can also be completed to determine if

MCS1 is valid on its own. Regardless of the approach, a future study is necessary to ensure the validity of the MCS1 assessment.

MCS1 can also be expanded upon to include more MATLAB-specific topics. Currently, MCS1 does not assess important concepts such as graphing or matrix manipulation. MATLAB is a unique programming language with specific uses unlike those of Java or C. As a result, MCS1 and the FYE program at Ohio State would likely benefit from additional questions that explicitly test these MATLAB-specific features.

Another possible future project following the development of MCS1 is to incorporate the assessment into the FYE program at Ohio State. MCS1 can serve as a benchmark or evaluation for students who have finished the first-year engineering course sequence. MCS1 can then be given again to upper-level students across the College of Engineering to determine in which areas students need reinforcement. A long-term project which compares scores on MCS1 taken by a group of students at the end of their first year and then again by the same students in, say, their third year could also potentially provide information on student retention of basic computer science topics through their undergraduate careers. The data gained from these projects could then be used to direct learning modules or curricula adjustments in upper-level courses to reinforce concept areas in which students score poorly.

Finally, MCS1, if validated, can be used as a starting point to develop more computer science concept inventories. These new concept inventories, in languages like C or Java, can then be used in the Computer Science and Engineering Department at Ohio State to evaluate the teaching methods and course curricula for the foundational programming courses taught to first and second year computer science students. Computer science has relatively few concept inventories and will be benefited by future projects that expand the breadth of concept inventories available to computer science educators [33].

References

[1] A. E. Tew, "Assessing Fundamental Introductory Computing Knowledge in a Language Independent Manner," 2010.

[2] A. E. Tew and M. Guzdial, "The FCS1: A Language Independent Assessment of CS1 Knowledge," in *SIGCSE '11 Proceedings of the 42nd ACM Technical Symposium on Computer Science Education*, Dallas, 2011.

[3] J. Libarkin, "Concept Inventories in Higher Education Science," *STEM Education Workshop 2,* pp. 1-10, 2008.

[4] E. Bardar, E. Prather and T. Slater, "Development and Validation of the Light and Spectroscopy Concept Inventory," *Astronomy Education Review,* vol. 5, no. 10, 2006.

[5] D. Hestenes, M. Wells and G. Swachamer, "Force Concept Inventory," *The Physics Teacher,* vol. 30, no. 3, 1992.

[6] M. W. Klymkowsky, S. M. Underwood and R. K. Garvin-Doxas, "Biological Concepts Instrument (BCI): A diagnostic tool for revealing student thinking," 2010.

[7] J. C. Libarkin, S. W. Anderson, D. Deeds and B. Callen, "Development of the geoscience concept inventory," *Proceedings of the National STEM Assessment Conference,* pp. 148-158, 2006.

[8] J. I. Smith and K. Tanner, "The Problem of Revealing How Students Think: Concept Inventories and Beyond," *CBE - Life Sciences Education,* vol. 9, no. 1, 2017.

[9] C. Taylor, D. Zingaro, L. Porter, K. Webb, C. Lee and M. Clancy, "Computer science concept inventories: past and future," *Computer Science Education,* pp. 1-24, 2014.

[10] L. Porter, D. Zingaro, S. N. Liao, C. Taylor, K. C. Webb, C. Lee and M. Clancy, "BDSI: A Validated Concept Inventory for Basic Data Structures," in *Proceedings of the 2019 ACM Conference on International Computing Education Research*, Toronto, 2019.

[11] D. Sands, M. Parker, H. Hedgeland, S. Jordan and R. Galloway, "Using concept inventories to measure understanding," *Higher Education Pedagogies,* vol. 3, no. 1, pp. 173-182, 2018.

[12] M. C. Parker, M. Guzdial and S. Engleman, "Replication, Validation, and Use of a Language Independent CS1 Knowledge Assessment," in *ICER '16 Proceedings of the 2016 ACM Conference on International Computing Education*, Melbourne, 2016.

[13] G. L. Herman, C. Zilles and M. C. Loui, "A psychometric evaluation of the digital logic concept inventory," *Computer Science Education,* vol. 24, no. 4, pp. 277-303, 2014.

[14] H. Danielsiek, W. Paul and J. Vahrenhold, "Detecting and Understanding Students' Misconceptions Related to Algorithms and Data Structures," in *Proceedigns of the 43rd ACM Technical Symposium on Computer Science Education*, Raleigh, 2012.

[15] M. F. Farghally, K. H. Koh, J. V. Ernst and C. A. Saffer, "Towards a Concept Inventory for Algorithm Analysis Topics," in *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, Seattle, 2017.

[16] S. Hamouda, S. H. Edwards, H. G. Elmongui, J. V. Ernst and C. A. Shaffer, "A basic recursion concept inventory," *Computer Science Education,* vol. 27, no. 2, pp. 121-148, 2017.

[17] K. Karpierz and S. A. Wolfman, "Misconceptions and Concept Inventory Questions for Binary Search Trees and Hash Tables," in *Proceedings of the 45th ACM Tehcnical Symposium on Computer Science Education*, Atlanta, 2014.

[18] L. Porter, S. Garcia, H.-W. Tseng and D. Zingaro, "Evaluating Student Understanding of Core Concepts in Computer Architecture," in *Proceedings of the 18th ACM conference on Innovation and technology in computer science education*, Canterbury, 2013.

[19] K. Webb and T. Cynthia, "Developing a pre- And post-course concept inventory to gauge operation systems learning," in *SIGCSE 2014 - Proceedings of the 45th ACM Technical Symposium on Computer Science Education*, 2014.

[20] D. Evans, G. L. Gray, S. Krause, J. Martin, C. Midkiff, B. M. Notaros, M. Pavelich, D. Rancour, T. Reed-Rhoads, P. Steif, R. Streveler and K. Wage, "Progress on Concept Inventory Assessment Tools," in *33rd ASEE/IEEE Frontiers in Education Conference*, Boulder, 2003.

[21] V. L. Almstrum, P. B. Henderson, V. Harvey, C. Heeren, W. Marion, C. Riedesel, L.-K. Soh and A. E. Tew, "Concept Inventories in Computer Science for the Topic Discrete Mathematics," in *ITiCSE*, Bologna, 2006.

[22] B. Hume and S. Sanders, "Enrollment Report 2019," Columbus, 2019.

[23] "2019 College of Engineering Annual Statistical Report," The Ohio State University, Columbus, 2019.

[24] Department of Engineering Education, "First-Year Engineering Program," The Ohio State University, [Online]. Available: https://eed.osu.edu/first-year-engineering-program.

[25] D. Morrell, "Design Of An Introductory Matlab Course for Freshman Engineering Students," in *2007 Annual Conference & Exposition*, Honolulu, 2007.

[26] A. Azemi and L. L. Pauley, "Teaching the introductory computer programming course for engineers using Matlab," in *2008 38th Annual Frontiers in Education Conference*, Saratoga Springs, NY, 2008.

[27] M. Wirth and P. Kovesi, "MATLAB as an Introductory Programming Language," *Computer Applications in Engineering Education,* vol. 14, no. 1, pp. 20-30, 19 April 2006.

[28] A. Azemi and L. Pauley, "Using Matlab to Teach the Introductory Computer-Programming Course for Engineers," in *American Society for Engineering Education Annual Conference & Exposition*, 2004.

[29] E. Charters, "The Use of Think-aloud Methods in Qualitative Research: An Introduction to Think-aloud Methods," *Brock Education,* vol. 12, no. 2, pp. 68-82, 2003.

[30] K. A. Neuendorf, The Content Analysis Guidebook, Thousand Oaks, California: Sage Publications, Inc., 2002.

[31] K. Krippendorff, Content Analysis: An Introduction to Its Methodology, 2nd ed., Thousand Oaks, California: Sage Publications, 2004.

[32] "Practical procedures in obtaining think aloud protocols," in *The Think Aloud Method: A practical guide to modelling cognitive processes*, London, Academic Press, 1994, pp. 44-47.

[33] A. Yadav, D. Burkhart, D. Moix, E. Snow, P. Bandaru and L. Clayborn, "Sowing the Seeds of Assessment Literacy in Secondary Computer Science Education: A Landscape Study," 07 2015.

[34] "2018 College of Engineering Annual Statistical Report," The Ohio State University, 2018.

[35] I. A. Halloun and D. Hestenes, "Common sense concepts about motion," *American Journal of Physics,* vol. 53, no. 11, pp. 1056-1065, 1985.

[36] Department of Engineering Education, "ENGR 1281.01H," The Ohio State University, [Online]. Available: https://eed.osu.edu/engr-1281.01h.

Appendix A: Programming Experience Questions

(1) Is this your first programming experience? This could include a previous programming workshop, computing course, functions in Excel, etc.
- Definitely yes
- Probably yes
- Might or might not
- Probably not
- Definitely not

(2) How would you identify your programming skills?
- I have no programming skills
- I have very little programming skills
- I have some programming skills
- I have strong programming skills
- I have very strong programming skills

(3) What has been your previous programming or computer science experience(s)? Please select all that apply.
- A computer science course at a high school
- A computer science course at a college
- A workshop or professional development session
- Programming utility tools (Excel, calculators, etc.)
- Java Script for web design
- Java Script for projects other than web design
- Self-taught via available resources
- Other (please specify)

(4) What programming languages (if any) do you consider yourself minimally proficient? Please select all that apply.
- Python
- Java
- Scratch
- C++
- C
- Racket/Scheme
- Jython
- Java Script
- Alice

- Jeroo
- LightBot
- Snap
- Bootstrap
- Processing
- Visual Basic
- None
- Other (please specify)

(5) Have you used any of the following:
- Code.org
- Codecademy
- Khan Academy
- Udacity
- None
- Other (please specify)

(6) Please select the option that best represents how you feel for each statement

	Strong disagree	Disagree	Neither agree nor disagree	Agree	Strongly Agree
I would like to take more courses in computing.	○	○	○	○	○
The skills in this class will be useful in my life.	○	○	○	○	○
The skills in this class will be useful in my career.	○	○	○	○	○
I know how to use programming to communicate with others.	○	○	○	○	○
I know how to use programming to communicate with programmers.	○	○	○	○	○

(7) How many times have you seen this assessment before?
- I haven't seen it before
- 1
- 2
- 3-5
- 6-9
- 10+

Appendix B. Think-Aloud Interview Revisions

#	Revision Category	Issue	# Participants Reporting	Resolved?	Notes
1	Question Revision	Removed "at the beginning" from choice A	2	Yes	
2					
3					
4	Question Revision	Update to have E be the correct answer	4	Yes	
4	Question Revision	Remove "of 4 characters" in code comment.	Researcher	Yes	
4	Typo	Make "not" in question bold and italicized to match other questions	Researcher	Yes	
5	Question Revision	Inconsistency between double quotes in question vs. single quotes in the function definition	1	Yes	
5	Typo	Removed parentheses in if-loop to be more consistent with other questions	Researcher	Yes	
5	Formatting	Change "concatentation" in question to be default font.	Researcher	Yes	
6	Question Revision	Conditional operators is defined in SCS1 pseudo code but not in MCS1	3	Yes	Add description in question
7	Question Revision	Added explanation of strcat from Q5 since questions will be randomized.	Researcher	Yes	
7	Question Revision	Change "the decoder function" in the question to the "secret function" to match function name.	1	Yes	
7	Question Revision	Every other letter can be interpreted to mean all letters	1	Yes	
8	Formatting	Format of answers should include the line breaks	2	Yes	Only applicable to paper exam.
9	Question Revision	Updated comment % rem(x,y) returns the remainder of x,y to read %rem(x,y) returns the remainder of the division of x/y	1	Yes	
10					
11					
12					
12	Answer Revision	Updated choice B to be incorrect so choice A is the only correct answer. Choice B was delta, alpha, bravo, charlie, echo and is now bravo, delta, alpha, echo, charlie.	2	Yes	
13					
14	Typo	Updated function [% missing code] = compress(s) to be function [abbr] = compress(s)	1	Yes	
14	Typo	Updated final if-else block to be inside the function	Researcher	Yes	

Categories
Answer Revision
Question Revision
Formatting
Typo

14	Answer Revision	Updated choices to only have one correct answer - E. Looks like A, B, and E are correct. A - changed missing code 1 length(abbr) ~= 0 to be length(abbr) == 0. B - changed missing code 1 length(abbr) == 0 to be length(abbr) ~= 0. Look at SCS1 choices A and B...not sure why they are incorrect	1	Yes	
14	Typo	Updated line 10 of code to use s(i) instead of s[i] and added a semicolon	Researcher	Yes	
14	Typo	Removed parentheses from line 11 of code	Researcher	Yes	
14	Question Revision	Updated question reference to call compress("Is this readable?") to compress('Is this readable?') to be consistent with MATLAB quotes	1	Yes	
14	Formatting	Indentations within the for loop should be consistent with the rest of the test	1	Yes	Only applicable to paper exam.
14					
14	Question Revision	Question mark included in the example, as well as the length(s-1) is inconsistent with description of the code	1	Yes	Make consistent with description
16	Question Revision	Added explanation of strcat from Q5 since questions will be randomized.	Researcher	Yes	
16	Answer Revision	charList(i) should be charList(k), i is not a variable in the code	3	Yes	
17					
18	Formatting	the "a" in answer b is in the incorrect font for code	1	Yes	Only applicable to paper exam.
19	Formatting	Font of condition 1, 2, 3 is incorrect. Should be code font	Researcher	Yes	Only applicable to paper exam.
19	Formatting	Answer choices split across pages was confusing	1	Yes	Only applicable to paper exam.
20					
21					
22					
23					
24	Question Revision	Added explanation of strcat from Q5 since questions will be randomized.	1	Yes	
24	Typo	Answer choice f should be e to be consistent with other choices	1	Yes	Applicable only to paper exam
24	Typo	Missing parenthesis after compute(x,y	3	Yes	Typo not present in Qualtrics